Management of Presentation Time in a Digital Media

Presentation System with Variable Rate Presentation Capability

[0001]   This application claims the benefit under 35 U.S.C. 119(e) of U.S. Provisional Application No. 60/255,226, filed on December 12, 2000.

Technical Field of the Invention

[0002]   One or more embodiments of the present invention pertain to the field of content presentation by a digital rendering system such as, for example, and without limitation, a digital media player.

Background of the Invention

[0003]   Most traditional digital rendering systems, such as RealNetworks® RealPlayer® digital media players, maintain an internal variable during playback of media content that reflects a current presentation time (hereafter referred to as "Current Time"). Current Time is, in effect, a current "position" in the media content that is being displayed and rendered. Typically Current Time is set to zero at the beginning of the media content, and it reaches a measure of time equal to a duration of presentation of the content of the entire work when the end of the media content is reached.

[0004]   In most traditional players, such as the RealPlayer® digital media player, a Current Time value is: (a) regularly calculated by a single module; (b) acquired and stored by core routines of the player; and (c) distributed to, and utilized by, various internal player objects. These internal objects utilize the Current Time value to determine when it is time to initiate or terminate various tasks associated with media content playback. The calculation of a Current Time value by the single module, and the distribution to, and utilization by, multiple objects within a player of the same Current Time value has a desirable result of keeping all objects synchronized.

[0005]   Typically the Current Time value must be regularly and accurately updated by the player, or the presentation of media content will be faulty. For instance, if the Current Time value is not updated sufficiently often, a video component of a media stream may appear uneven or jumpy, and gaps in an audio content may be audible.

[0006]   Although the concept of Current Time seems straightforward, in fact, it conflates two subtly different properties of media playback. The first property of media playback that is conflated in the concept of Current Time is a time elapsed since the beginning of the media content presentation (hereafter called "Presentation Time"). Thus, if the media has been playing for one minute, the value of Presentation Time is 60,000 milliseconds. All of time values discussed herein can be measured in various units. Two popular units are milliseconds, and centi-nanoseconds, or 1/10,000,000 of a second. The unit of measurement is not an issue here. Other considerations of representing time that are not issues here are the precision, the range of values, and the format of the representation.

[0007]   The second property of media playback that is conflated in the concept of Current Time is a location in the media content stream that is currently being played (hereafter called "Content Time"). In a traditional linear media stream that is always played back at a fixed, "normal" rate, any given content element is always presented after a fixed amount of time has elapsed from the beginning of playback. Because of this, each such content element can be regarded as having a timestamp associated with it, i.e., a time value specifying how long it would take to reach that location, starting from the beginning of the media content, and playing at normal rate. Hereinafter we will call this property "Data Time."

[0008]   Presentation Time and Data Time are identical in traditional players, because traditional players can only present media content at a fixed "normal" rate. However, when a player is enhanced with a Time-Scale Modification (TSM) capability, it can present media content at various rates. Because of this, Presentation Time and Data Time are no longer the same. For example, if a 60-second clip of media content is presented at a fixed rate that is twice normal rate, at the end of the clip the Data Time is 60,000 milliseconds, but the Presentation Time is 30,000 milliseconds. This is because it only takes 30 seconds to play the 60-second clip.

[0009]   We have discovered that problems may occur when a traditional player is enhanced with TSM functionality. In particular, if a Current Time value is distributed to multiple objects, some of them may interpret the Current Time value as specifying Data

Time, some of them may interpret the Current Time value as specifying Presentation Time, and some of them may interpret the Current Time value as specifying both Data and Presentation Time. Thus, a first problem occurring when a traditional player is enhanced with TSM functionality is that the significance of the time value distributed to multiple objects is, in general, ambiguous. A second problem occurring when a traditional player is enhanced with TSM functionality is that Data Time does not, in general, equal Presentation Time, and the calculation, storage, and distribution of a single time value is inadequate to specify both values.

[00010] It is quite common for media players to rely on an audio renderer (for example, a player object responsible for outputting audio content through, for example, a computer sound card) to calculate and update the Current Time value. This is done because the nature of audio representation is such that typically each audio data element has either a fixed, or explicitly specified presentation duration, associated with it, and these presentation durations are enforced by audio rendering hardware. Therefore, the audio renderer can typically determine Presentation Time either by maintaining a running total of the presentation durations of all audio data elements rendered since playback began, or in some cases by querying the audio rendering hardware itself for the equivalent value.

[00011] If a media player does in fact acquire the Current Time value from the audio renderer, the value that the audio renderer will return to the system will typically be the Presentation Time. Since most of the rest of the system needs Data Time, most of the rest of the system can no longer employ the value returned by the audio renderer object.

[00012] As one can readily appreciate from the above, a need exists in the art for a method and apparatus for solving one or more of the above-described problems.

Summary of the Invention

[00013] One or more embodiments of the present invention advantageously satisfy one or more of the above-described problems. In particular, one or more embodiments of the present invention provide a method for managing Presentation Time in a digital rendering system for presentation of temporally-ordered data when the digital rendering system includes a Variable Rate Presentation capability. Specifically, one embodiment of

the present invention is a method for converting Presentation Time to Data Time, and for reporting Data Time instead of Presentation Time when only one time can be returned.

Brief Description of the Figure

[00014]   FIG. 1 shows a block diagram of a Presentation System embodied as a RealNetworks® RealPlayer® application running on a computer; and

[00015]   FIG. 2 shows a block diagram of a generalized Presentation system that includes Presentation System Controller Modules, Other Presentation Modules (including a Presentation Rate Modification Module), and a number of Renderers.

Detailed Description

[00016]   In accordance with one embodiment of the present invention, Presentation System 100 (a more general definition of a Presentation System is provided below) is embodied as a RealNetworks® RealPlayer® application running on a computer, for example, under some version of the Microsoft Windows operating system. As shown in FIG. 1, application module 110 sends to, and receives from, Player Core object 120, control and status messages such as, for example, Play, Pause, Stop, and so forth. Temporal Sequence Presentation Data, also referred to herein as Presentation Data, (a more general definition of Temporal Sequence Presentation Data is provided below) is embodied as streaming media content and is delivered to the RealPlayer® application over the Internet, a local-area network (LAN), or from files stored in the computer that is executing the RealPlayer® application. For example, in accordance with one embodiment, the Presentation Data, for example, audio data, is received by media content source module(s) 130, and are placed in audio media data buffers 140 that are made available to Player Core object 120.

[00017]   As will be defined in more detail below, each data element of the Presentation Data has a Rendition Type that corresponds to a type of Renderer (a more general definition of Renderer is provided below) that can be used to render the data element. For example, for the embodiment described above, the Rendition Types that can be rendered include but are not limited to audio (encoded in various formats), video, still images, text, and scripts. In particular, audio is a Time-Distinguished Rendition Type (a

more general definition of Time-Distinguished Rendition Type is provided below). As a result, for this embodiment, audio is organized within the RealPlayer® application in buffers that contain, for example, 100-milliseconds of sample values. Further, every buffer is timestamped, so these buffers are Timestamped Elements (as more generally described below, this means that the Data Time of the element is explicitly represented as part of the element), and the time associated with the first sample in every buffer is specified in milliseconds.

[00018]   In accordance with this embodiment, the Rendition Period (as more generally described below, this is the length of time the rendering process should last for the element) of the audio buffers is 100 milliseconds. In some ways the individual audio samples play the part of the data elements as described below. It would be obvious to someone of ordinary skill in the art how to sometimes regard 100 millisecond buffers of samples and sometimes the individual samples themselves as audio elements. In accordance with this embodiment, a sample period of the individual audio samples is stored in a header that is part of the sample buffer definition (for example, one $10,000^{th}$ of a second is a typical sample period).

[00019] .   In accordance with this embodiment, an object called TSMAudioDevice object 150 combines functions of the Renderer for audio data (TSMAudioDevice Audio Renderer 160) and a Variable Rate Presentation Module (a more general definition of Renderer is provided below) (TSMAudioDevice VRP Module 170). In accordance with this embodiment, the audio Renderer part of TSMAudioDevice object 150 (i.e., TSMAudioDevice Audio Renderer 160) is a Timing Renderer (a more general definition of a Timing Renderer is provided below) for Presentation System 100. Note that the RealNetworks® RealPlayer® application does not include support for variable rate playback. However, Plug-In 180 comprises a product called a 2xAV Plug-In is available from Enounce, Incorporated of Palo Alto, California. When the 2xAV Plug-In is installed on a computer that has had the RealPlayer® application previously installed, it "plugs into" the RealPlayer® application, and adds variable rate playback capability thereto. The 2xAV Plug-In has its own User Interface, which includes a slider that a user can manipulate to

adjust playback rate. In operation, the 2xAV Plug-In communicates with TSMAudioDevice object 150 by sending messages through an object called State Information Exchange Server 190 ("SIX Server 190").

[00020] Thus, in accordance with this embodiment, TSMAudioDevice object 150 accepts messages from SIX Server 190 that specify a desired playback or presentation rate. Playback or presentation rate values can range from 0.3 to 3.0 (a rate of 1.0 is normal; a rate of 0.3 is 30% of the normal rate; and a rate of 3.0 is three times faster than the normal speed). TSMAudioDevice object 150 receives SIXExchange messages from SIX Server 190, and stores a requested playback rate value contained in these messages as a new value of an internal Current Presentation Rate parameter or property. In addition, as shown in FIG. 1, TSMAudioDevice object 150 receives buffers 200 of audio data to be rendered (i.e., played out through the computer's sound card) from Player Core object 120. When TSMAudioDevice object 150 receives buffers 200 of audio data to be rendered, it is processed by TSMAudioDevice VRP Module 170. TSMAudioDevice VRP Module 170 processes buffers 200 through a library of signal processing routines, for example, a suitable library of signal processing routines called the Time Scale Tailor package is available from Enounce, Incorporated of Palo Alto, California. In accordance with this embodiment, this library carries out digital signal processing procedures on buffers 200 of audio samples that has the effect of reducing the number of samples in the buffer (when playing faster than real time) or increasing the number of samples in the buffer (when playing slower than real time), thereby effectively changing the playback rate. For example, in accordance with this embodiment, processing the buffer using the library decreases or increases the samples in a particular way so as to leave the perceptual and linguistic information in the buffers unchanged, but to change the duration of the buffers. Additionally, playback rate parameters, unmodified and modified buffer lengths and Rendering Period values, and other time-related values are calculated by TSMAudioDevice VRP Module 170, and are stored with each audio buffer. Then, modified audio data buffers 210 are stored by TSMAudioDevice VRP Module 170 for presentation by TSM AudioDevice Audio Renderer 160. TSM AudioDevice Audio Renderer 160 comprises

Audio Renderer Core Object 165 that submits modified buffers 210 for processing to the computer's audio electronics, for example, Computer Sound Card 220. For example, as shown in FIG. 1, Core Object 165 comprises an interface known as a WAV driver. This interface is defined by Microsoft, and is supported by the Windows operating system.

[00021]   On a regular basis during playback or presentation, Player Core object 120 calls a method implemented by TSMAudioDevice object 150 called GetCurrentAudioTime(). This method returns a Current Time in milliseconds. Additionally, every time a buffer of audio samples is processed (for example, buffer 200), TSMAudioDevice object 150 is responsible for calling a Player Core object 120 method called OnTimeSync(), and passing to the Player Core object 120 method the Current Time in milliseconds. Player Core object 120 interprets both of these times as Data Times. In this embodiment, Presentation System 100 (other than TSMAudioDevice object 150) does not support the concept of Presentation Times that are different than Data Times. To do this, in accordance with one embodiment of the present invention, TSMAudioDevice object 150 carries out methods described below to convert Presentation Time (for example, as reported by its WAV driver Core object routines) into Data Time (as needed by Player Core object 120).

[00022]   Before describing the methods to convert Presentation Time into Data Time, we present generalizations of the matters described above in conjunction with FIG. 2. In particular, FIG. 2 shows a block diagram of a generalized Presentation system that includes: Presentation System Controller Modules 400, Other Presentation Modules 410 (including a Presentation Rate Modification Module), and a number of Renderers, for example, Audio Renderer 420, Video Renderer 430, and Other Renderers 440. Further, as shown in FIG. 2, Temporal Sequence Presentation Data 450 is applied as input to Other Presentation Modules 410.

[00023]   As defined herein, a Presentation System means a system or method that: (a) acquires, interprets, decodes, and manages presentation of Temporal Sequence Presentation Data (defined below); (b) selects, instantiates, initializes, controls, and monitors Renderers (defined below); (c) initiates a presentation by determining which presentation data

elements are to be submitted first to which Renderers, effecting such submission, and causing the Renderers to begin processing; (d) maintains a Current Time parameter, whose value is regularly updated in a monotonically-increasing fashion during linear presentation (the value may be set to zero or any other value when presentation is stopped, or a jump is performed to a non-sequential location) --the Presentation System may maintain and update the value of the Current Time parameter by identifying a Renderer that can reliably maintain and update its Cumulative Rendition Period, and arrange to acquire the value of that parameter at regular intervals; (e) distributes its Current Time parameter to other Presentation Modules as needed; and (f) manages the presentation process, including by determining which data elements should be submitted next to which Renderers, and by comparing its Current Time value to the Data Time of those data elements, thereby determining when to effect such submission.

[00024]   A digital media player is a common type of Presentation System, but there are many other types of Presentation Systems. For example, a controller that processes a script which causes digitally-controlled manufacturing equipment to make a printed circuit board is also a Presentation System, as is a controller that processes a script which causes a robot to perform a dance.

[00025]   As defined herein, a Renderer is a system or method having the following characteristics: (a) the Renderer processes Temporal Sequence Presentation Data (defined below); (b) the Renderer processes data elements in an ordered sequence in which "earlier" elements are processed before "later" elements (the order may be determined by the order in which the elements are submitted to the Renderer, or by the Data Times (defined below) of the elements, or by using other techniques); (c) processing a data element takes a finite amount of time (possibly but not typically zero) known as the Rendition Period of the data element; (d) processing a sequence of data elements takes a finite amount of time directly related to the sum of the Rendition Periods of the individual elements, and, potentially, some other factors (the amount of time required to process (render) a sequence of data elements is called a Cumulative Rendition Period for those elements); and (e) at least one instance of a Renderer (often associated with rendering of audio data) has a capability of

reporting back to a module, for example, a Presentation System Control Module, upon request, a current value of the Cumulative Rendition Period (a Renderer that is consistently used by the Presentation System in this fashion is referred to as a Timing Renderer).

[00026]   As defined herein, Temporal Sequence Presentation Data, also referred to as Presentation Data, means data having the following characteristics: (a) the purpose, utility, or semantics of the data is closely associated with its presentation --presentation involves rendering of the data to achieve some effect (including but not limited to constituting a visible and/or audible presentation that can be monitored by a human being); (b) there are a plurality of rendering processes capable of effecting an appropriate presentation of the data; (c) the data comprises a set of elements; (d) each data element has a Rendition Type that corresponds to a type of Renderer that can be used to render the data element --some common Rendition Types are Pulse Code Modulation (PCM) audio, MPEG video, and JPEG images; (e) one or more Rendition Types may be Time-Distinguished Rendition Types --Time-Distinguished Rendition Types are Rendition Types of Temporal Sequence Presentation Data whose intrinsic characteristics and whose natural rendition process make them preferred candidates for defining and maintaining a system-wide Current Time parameter (note that most audio Rendition Types are Time-Distinguished Rendition Types); (f) associated with each element is a Data Time --the Data Time of some elements may be explicitly represented as part of the element (such elements are called Timestamped Elements), and the Data Time of some elements may be derivable only by performing or simulating an appropriate rendering process on all or part of the Presentation Data (such elements are called Sequential Elements); (g) the elements have a partial ordering, so that when performing rendering operations on the data it is possible to determine i) which data elements to deliver to the Renderers to begin the presentation process; and ii) given that the presentation process has reached a certain point, which data elements to deliver to the Renderers next to continue the presentation process; and (h) associated with each element is a Rendition Period -the Rendition Period is the length of time the rendering process should last for that element, where the Rendition Period of an element may be specified in many different ways, including but not limited to the following: (i) as a value explicitly

stored as part of the element, (ii) as a fixed value associated with that type of data element, and stored in a header field of the Presentation Data, (iii) as a fixed value associated with a Presentation System, (iv) a difference between the Data Time of the element and the Data Time of a following element that would be submitted to the same Renderer in the course of presentation (i.e., the element is rendered until there is another element to be rendered by the same Renderer), (v) as a fixed property of the rendering process.

[00027]   As defined herein, a Variable Rate Presentation ("VRP") Module (also known as a VRP Module) means a module that: (a) accepts control commands and messages from the Presentation System; (b) maintains and, in response to commands from the Presentation System, updates the value of a Current Presentation Rate parameter where values of this parameter have the following interpretation: (i) a value of 1.0 means that presentation is to take place at the "normal" or default rate, (ii) a value less than 1.0 but greater than zero means that presentation is to take place slower than "normal" (the factor by which presentation is to be slowed down is equal to the inverse of the Current Presentation Rate value), (iii) a value greater than 1.0 means that presentation is to take place faster than "normal" (the factor by which presentation is to be sped up is equal to the Current Presentation Rate value), and (iv) a value less than zero is interpreted identically to the corresponding positive value of the parameter, but the direction of presentation is reversed (i.e., the presentation "runs in reverse"); (c) processes Temporal Sequence Presentation Data; (d) modifies Temporal Sequence Presentation Data in a manner consistent with the value of the Current Presentation Rate parameter and the Renderers to which the data will be later submitted, having the effect that the rate with which processing takes places will track the value of the Current Presentation Rate parameter.   The implementation of Variable Rate Presentation may also involve one or more Renderers. In this case the value of the Current Presentation Rate parameter is attached to the data elements, or otherwise communicated to the appropriate Renderers.

[00028]   In a Presentation System fabricated in accordance with one embodiment of the present invention, the Presentation System would maintain two separate values of the Current Time parameter. The first of the values would be a Current Data Time value that

indicates the largest Data Time value of any data element that has already been submitted for rendering, or should have already been submitted for rendering. The second of the values would be a Current Presentation Time value, which would be the Cumulative Rendition Period of all data elements submitted since presentation began (i.e., the elapsed rendering time). In a Presentation System fabricated in accordance with another embodiment of the present invention, the Presentation System would maintain a single value of the Current Time parameter. Such an embodiment is typical of Presentation Systems that were not designed with the notion of variable rate playback in mind. More specifically, some such systems were designed with an implicit assumption that the only possible presentation rate was 1.0.

[00029]   Presentation Time and Data Time are identical properties in traditional Presentation Systems, such as media players. However, when a Presentation System is enhanced with a Variable Rate Presentation ("VRP") capability, these properties are no longer the same. We have discovered that this presents a problem when a traditional media player or other Presentation System is enhanced with a VRP capability, for two reasons. First, if a Presentation System Control Module only acquires a single Current Time value from a Timing Renderer, it is probably assuming that that value can be interpreted as both Current Data Time and Current Presentation Time. If these times are not equal, at least one of those assumptions will be in error. Secondly, if a single Current Time value is distributed to multiple components, some of which interpret the value as Current Data Time, and some of which interpret the value as Current Presentation Time, at least one of these interpretations will be in error. We have invented a way to convert Presentation Time to Data Time, and we have invented a method for reporting Presentation Time when only one time can be returned.

[00030]   In accordance with one embodiment of the present invention, certain time-related properties (that will later be used to calculate Current Presentation Time and Current Data Time) are associated with Temporal Sequence Presentation Data elements by taking the following steps.

[00031]   Step 1: from time to time as a user or some other controlling entity decides to change a rate of presentation, the Presentation System Control Module sends a message to a Variable Rate Presentation Module (VRP Module), specifying an updated value for a Current Presentation Rate parameter. When the VRP Module receives this message, it updates the value of its Current Presentation Rate parameter.

[00032]   Step 2: in preparation for presentation, the Presentation System organizes Temporal Sequence Presentation Data elements (for example, audio samples) into collections called buffers.

[00033]   Step 3: buffers are presented in an ordered or semi-order fashion for Presentation Rate modification and rendering, typically according to the Data Time of the first data element in a buffer.

[00034]   Step 4: the number of unmodified samples in each buffer is determined, and the Unmodified Rendition Period of each element is obtained. The value of the Current Presentation Rate parameter is held constant (i.e., it is not allowed to change) while the VRP Module is processing a buffer. Also, the Rendition Period of all data elements in a buffer is constrained to be equal. Note, however, if it were desired to vary either the Current Presentation Rate or the Rendition Period within a buffer, that buffer could be broken down into multiple smaller buffers in which those properties were constant. In doing so, if necessary, each buffer could hold only a single data element.

[00035]   Step 5: the Unmodified Cumulative Rendition Period for the buffer is calculated and retained as a property of the buffer by multiplying the Rendition Period of each data element in the buffer by the number of unmodified data elements in the buffer.

[00036]   Step 6: the Data Time of the buffer is defined to be the Data Time associated with the first unmodified data element in the current buffer. The Data Time is acquired or calculated, and retained as a property of the buffer. If it is not directly specified as a property of the first data element of the current buffer, it can be calculated by adding the Data Time of the previous buffer to the Unmodified Cumulative Rendition Period of the previous buffer.

[00037]   Step 7: the data elements in the current buffer are presentation rate modified, so that the ratio of the Cumulative Rendition Period of the buffer prior to presentation rate modification, divided by the Cumulative Rendition Period of the buffer following modification, is substantially equal to the Current Presentation Rate.

[00038]   Step 8: the number of modified data elements in the modified buffer, and the Modified Rendition Period of each data element in the buffer, is determined and retained as a property of the buffer.

[00039]   Step 9: the Modified Cumulative Rendition Period for the buffer is calculated and retained as a property of the buffer by multiplying the Modified Rendition Period of each data element in the buffer by the number of modified data elements in the buffer.

[00040]   Step 10: the Modified Presentation Time of the buffer is defined to be the Presentation Time associated with the first modified data element in the buffer. This time is calculated and retained as a property of the buffer. It is calculated by adding to the Modified Presentation Time of the first modified data element of the previous buffer, the Modified Cumulative Rendition Time of the previous buffer.

[00041]   Step 11: calculate, and retain as a property of the current buffer, the Cumulative Modified Data Element Count associated with the first data element in the current buffer by adding the Cumulative Modified Data Element Count associated with the first modified data element in the previous buffer to the number of modified data elements in the previous buffer.

[00042]   Note that in this embodiment only the VRP Module needs to be informed of the current value of the Presentation Rate parameter. Renderer Modules, on the other hand, get all of their information about Presentation Rate from the buffer properties described above.

[00043]   In accordance with one embodiment of the present invention, each Renderer is assumed to comprise a Core component. This Core component may be hardware, software, or a combination of hardware and software. For example, the Core component of an audio Renderer may be a Sound Card and its associated driver software. The Core

component performs the essential rendering process for the particular Type of Temporal Sequence Presentation Data that the Renderer processes.

[00044] In accordance with one embodiment of the present invention, the Core component can be asked at any point in time to report the number of data elements rendered since a distinguished event such as, for example, an invocation of an Initialize or Clear command. Equivalently, the Core component rendering hardware or software may be able to report the number of milliseconds of rendering that has occurred since a distinguished event.

[00045] Renderers, especially Timing Renderers, must decide how to respond when other components of the Presentation System ask for the Current Time value without specifying whether Presentation Time or Data Time is desired. In many cases it is possible to determine that the Presentation System really wants to know what the Current Data Time is. Therefore in accordance with one embodiment of the present invention, a certain Data Time is returned when a request is made for the Current Time. For this, and other reasons, Renderers, especially Timing Renderers, must maintain an up-to-date and accurate value for both the Presentation Time and the Data Time associated with the data element currently being rendered. In accordance with one embodiment of the present invention, it is the Data Time of the data element currently being rendered by the Core component that is returned as the Current Time when the Current Time is requested by another module. Therefore, in accordance with one embodiment of the present invention, Current Presentation Time and Current Data Time are calculated by taking the following steps.

[00046] Step 1: a modified buffer with its associated properties as described above is submitted to an appropriate Renderer.

[00047] Step 2: if the Renderer's Core component is capable of reporting the number of milliseconds of rendering that has occurred since a distinguished event, and the Modified Presentation Time of this modified buffer is zero (or some other distinguished value), the Renderer triggers the distinguished event in its Core component.

[00048] Step 3: the Renderer submits the contents of the buffer to its Core component.

[00049]   Step 4: the Renderer also stores each modified buffer in some method that allows ready access to all of the buffer properties until it has determined that all data elements in the buffer have been rendered.

[00050]   Step 5: if the Core component is capable of reporting the number of data elements rendered since a distinguished event occurred, the Renderer calculates the Current Data Time and the Current Presentation Time using the following algorithm.

[00051]   Step 5a: it asks its Core component for the cumulative number of data elements rendered.

[00052]   Step 5b: it determines which buffer the next data element to be rendered will have come from, by identifying the particular buffer that has for its Cumulative Modified Data Element Count the largest cumulative sample number less than or equal to the reported number of data elements rendered --this buffer is referred to as the current rendering buffer.

[00053]   Step 5c: the current Data Time is equal to the Data Time of the current rendering buffer, plus an offset.

[00054]   Step 5d: the offset is calculated by multiplying the unmodified buffer duration by the Completion Fraction.

[00055]   Step 5e: the Completion Fraction is calculated by subtracting the cumulative sample number associated with the first sample in the current rendering buffer from the Core component's reported number, and dividing the result by the number of modified samples in the buffer.

[00056]   Step 5f: the Current Presentation Time is equal to the Modified Presentation Time of the current rendering buffer, plus an offset.

[00057]   Step 5g: the offset is calculated by multiplying the buffer's Modified Cumulative Rendition Period by the Completion Fraction.

[00058]   Step 6: if the Core is capable of reporting the number of milliseconds of rendering that has occurred since a distinguished event, the Renderer calculates the Current Data Time and the Current Presentation Time using the following algorithm.

[00059]   Step 6a: it asks its Core component for the number of milliseconds of rendering that has occurred.

[00060]   Step 6b: it determines which buffer the next data element to be rendered will have come from by identifying the particular audio buffer that has for its Modified Presentation Time the largest value less than or equal to the Core's reported value --this buffer is referred to as the current rendering buffer.

[00061]   Step 6c: the current Data Time is equal to the Data Time of the current rendering buffer, plus an offset.

[00062]   Step 6d: the offset is calculated by multiplying the unmodified buffer duration by the Completion Fraction.

[00063]   Step 6e: the Completion Fraction is calculated by subtracting the Modified Presentation Time of the current rendering buffer from the Core component's reported value, and dividing the result by the Cumulative Modified Rendering Period of the buffer.

[00064]   Step 6f: the Current Presentation Time is equal to the Modified Presentation Time of the current rendering buffer, plus an offset.

[00065]   Step 6g: the offset is calculated by subtracting the Modified Presentation Time of the current rendering buffer from the Core component's reported value.

[00066]   Step 6h: the current Data Time is reported to the player as the Presentation Time.

[00067]   Step 7: whenever an object requests the Current Time, the Renderer computes an updated value for the Presentation Time and Data Time, and reports either or both times as appropriate.

[00068]   Those skilled in the art will recognize that the foregoing description has been presented for the sake of illustration and description only. As such, it is not intended to be exhaustive or to limit the invention to the precise form disclosed.